

LECTURE 9 : OPTICAL FLOW

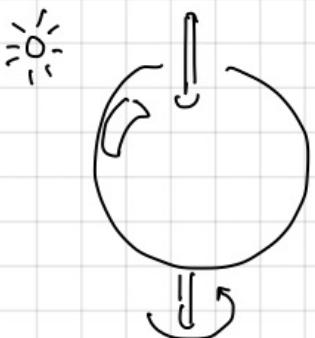
What is Optical flow?

The apparent motion of
brightness patterns

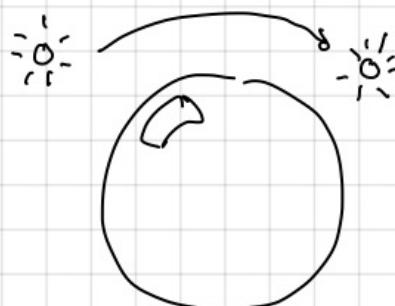
If consists of motion direction
& magnitude

It's not the same as Motion
flow (= the actual motion/velocity
vector in the real world)

The differences between optical flow & motion flow:

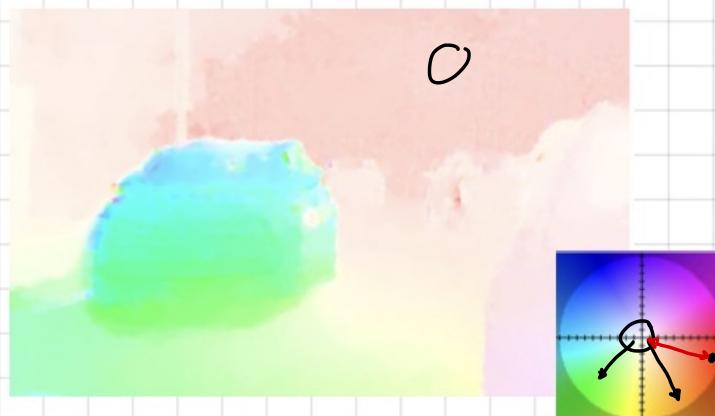


Non-zero motion field
Zero optical flow



Nonzero optical flow
Zero motion field

Many applications of optical flow: object tracking, motion analysis, collision detection, image registration, obstruction detection, motion segmentation, etc.



[2] Lucas-Kanade Optic Flow

Brightness Constancy Constraint (BCC) : common assumption

The brightness of pixels representing an object is constant throughout frames.

Intensity/brightness of a pixel : $I(x, y, t)$

Under the BCC :

$$I(x, y, t) = I(x+u, y+v, t+r)$$

where : (u, v) is the optic flow vector
 t indicates the frame index

Applying the chain rule :

$$\frac{dI(x, y, t)}{dt} = \frac{dI}{dx} \frac{dx}{dt} + \frac{dI}{dy} \frac{dy}{dt} + \frac{dI}{dt} = 0$$

$$u = \frac{dx}{dt} ; v = \frac{dy}{dt}$$

$$\text{in discrete: } \frac{\Delta x}{\Delta t} = \frac{x_{t+r} - x_t}{(t+r) - t} = \frac{(x_t + u) - x_t}{r} = u$$

Hence:

$$I_x u + I_y v + I_t = 0 \quad \text{the BCC}$$

Lucas-Kanade's objective function:

$$\min_{\{u, v\}} E(u, v) = \min_{\{u, v\}} (I_x u + I_y v + I_t)^2$$

Minimization:

$$\textcircled{1} \quad \frac{\partial E(u, v)}{\partial u} = 2(I_x u + I_y v + I_t) I_x = 0$$

$$\textcircled{2} \quad \frac{\partial E(u, v)}{\partial v} = 2(I_x u + I_y v + I_t) I_y = 0$$

$$\begin{bmatrix} I_x I_x & I_y I_x \\ I_x I_y & I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_x I_t \\ I_y I_t \end{bmatrix}$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} I_x I_x & I_y I_x \\ I_x I_y & I_y I_y \end{bmatrix}}_{A^{-1}}^{-1} \begin{bmatrix} I_x I_t \\ I_y I_t \end{bmatrix}$$

To be invertible, $\det(A) \neq 0$

$$\det(A) = I_x^2 I_y^2 - (I_x I_y)(I_y I_x) = 0 \rightarrow A \text{ not invertible}$$

Solution: Instead a pixel operation, we employ a patch operation
(e.g. 3×3 , 5×5 , 7×7 , etc)

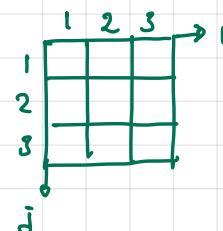
Considering a 3×3 patch:

$$I_{x_{1,1}}^2 u + I_{y_{1,1}} I_{x_{1,1}} v = - I_{t_{1,1}} I_{x_{1,1}}$$

$$I_{x_{1,2}}^2 u + I_{y_{1,2}} I_{x_{1,2}} v = - I_{t_{1,2}} I_{x_{1,2}}$$

$$\vdots \quad \vdots \quad \vdots$$

$$I_{x_{3,3}}^2 u + I_{y_{3,3}} I_{x_{3,3}} v = - I_{t_{3,3}} I_{x_{3,3}}$$



$$\sum_{i,j} I_{x_{ij}}^2 u + \sum_{i,j} I_{y_{ij}} I_{x_{ij}} v = - \sum_{i,j} I_{t_{ij}} I_{x_{ij}} +$$

$$\sum_{i,j} I_{x_{ij}}^2 u + \sum_{i,j} I_{y_{ij}} I_{x_{ij}} v = - \sum_{i,j} I_{t_{ij}} I_{x_{ij}}$$

Finally:

$$\begin{bmatrix} \sum_{i,j} I_{x,i,j}^2 & \sum_{i,j} I_{x,i,j} I_{y,i,j} \\ \sum_{i,j} I_{y,i,j} I_{x,i,j} & \sum_{i,j} I_{y,i,j}^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -\sum_{i,j} I_{\epsilon,i,j} I_{x,i,j} \\ -\sum_{i,j} I_{\epsilon,i,j} I_{y,i,j} \end{bmatrix}$$

Based on the equation, (u, v) is sparse & local flow field;
Since for many regions are textureless:

$$\sum_{i,j} I_{x,i,j}^2 = \sum_{i,j} I_{y,i,j}^2 = \sum_{i,j} I_{y,i,j} I_{x,i,j} = 0$$

In practice: $\begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} G * I_x^2 & G * I_x I_y \\ G * I_x I_y & G * I_y^2 \end{bmatrix}^{-1} \begin{bmatrix} G * I_x I_\epsilon \\ G * I_y I_\epsilon \end{bmatrix}$

Note: G is a Gaussian kernel with a certain window size (e.g. 5×5),
where $(G * I_x^2)(G * I_y^2) \neq (G * I_x I_y)^2$

Example:



Q: How to compute a dense & global optic flow?
 ↓
 (every pixel)

[3] Horn - Schunck Optical Flow

Objective function:

$$E(u,v) = \underbrace{(I_x u + I_y v + I_t)^2}_{\text{data term}} + \alpha \underbrace{\left(\|\nabla U\|^2 + \|\nabla V\|^2 \right)}_{\text{smoothness prior term}}$$

where : $\nabla U = \begin{bmatrix} \partial U / \partial x \\ \partial U / \partial y \end{bmatrix} = \begin{bmatrix} U_x \\ U_y \end{bmatrix}$

$$\|\nabla U\|^2 = \left(\sqrt{U_x^2 + U_y^2} \right)^2 = U_x^2 + U_y^2$$

Global objective function: $J(u,v) = \int E(u,v) dx dy = 0$

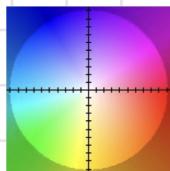
Goal: To find (u,v) for every pixel that minimizes the entire image.

This is exactly the same problem studied in calculus of variations:
a field dealing with optimizing a functional (= a function of a function).
 $E(u,v)$ is a function of (u,v) functions.

One of the methods to solve the global minimization:

Euler - Lagrange Equation

Example of dense flow:



[4] Euler - Lagrange Equation

#6

General form: $J = \int F(y, y', t) dt$; $y' = dy/dt$

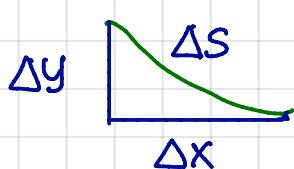
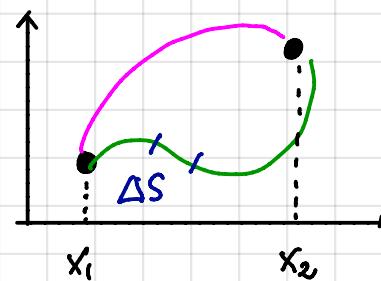
J will have a stationary value if : $\frac{\partial F}{\partial y} - \frac{\partial}{\partial t} \left(\frac{\partial F}{\partial y'} \right) = 0$

Euler-Lagrange equation is key in calculus of variations.

Example:

The shortest distance between 2 points :

(We know it will be a straight line,
but how can we prove it?)



$$\Delta S \cong \sqrt{\Delta x^2 + \Delta y^2}$$

$$ds = \sqrt{dx^2 + dy^2} \rightarrow ds = \frac{dx}{\sqrt{dx^2 + dy^2}}$$

$$s = \int \sqrt{1 + (y')^2} dx \quad \leftarrow \quad \int ds = \int \sqrt{1 + \left(\frac{dy}{dx}\right)^2} dx$$

$$\text{To minimize : } s = \int F(x, y') dx ; \quad F = \sqrt{1 + (y')^2}$$

where according to the Euler-Lagrange eq. :

$$\frac{\partial F}{\partial y} - \frac{\partial}{\partial x} \left(\frac{\partial F}{\partial y'} \right) = 0$$

Goal: to find y
that minimizes s
 $\min_{\{y\}} s = \min_{\{y\}} \int F(x, y') dx$

(1) $\frac{\partial F}{\partial y} = 0$, since function S is independent from y .

$$(2) \frac{\partial F}{\partial y'} = \frac{d}{dy'} \left(\sqrt{1 + (y')^2} \right) = \frac{1}{2} (1 + y')^{-1/2} \cdot 2y' = \frac{y'}{\sqrt{1 + (y')^2}}$$

Hence : $\frac{\partial}{\partial x} \left(\frac{y'}{\sqrt{1+(y')^2}} \right) = 0$

Recall : Ordinary Differential Equation :

$$\frac{dy}{dx} = 0$$

$$\int \frac{dy}{dx} dx = \int 0 dx$$

$$\int dy = C_1$$

$$y + C_2 = C_1$$

$$y = C$$

because the derivative
of C w.r.t. x is 0

Therefore :

$$\frac{\partial}{\partial x} \left(\frac{y'}{\sqrt{1+(y')^2}} \right) = 0 \rightarrow \frac{y'}{\sqrt{1+(y')^2}} = C$$

$$y' = C \sqrt{1+(y')^2} = \sqrt{C^2 + C^2(y')^2}$$

$$(y')^2 = C + C(y')^2 ; \text{ whether it's } C \text{ or } C^2, \\ (y')^2 (1-C) = C \quad \text{it doesn't matter}$$

$$y' = \frac{dy}{dx} = \sqrt{\frac{C}{1-C}} = C$$

$$\int \frac{dy}{dx} dx = \int C dx$$

$$y = Cx + b \rightarrow \text{It's indeed a straight line.}$$

[5] HS Global Optimization

#8

Global objective function: $J(u, v) = \int E(u, v) dx dy$

where :

$$E(u, v) = (I_x u + I_y v + I_t)^2 + \alpha (\|\nabla u\|^2 + \|\nabla v\|^2)$$

Using the Euler - Lagrange equation :

$$\textcircled{1} \quad \frac{\partial E}{\partial u} - \frac{\partial}{\partial x} \frac{\partial E}{\partial u_x} - \frac{\partial}{\partial y} \frac{\partial E}{\partial u_y} = 0$$

$$\textcircled{2} \quad \frac{\partial E}{\partial v} - \frac{\partial}{\partial x} \frac{\partial E}{\partial v_x} - \frac{\partial}{\partial y} \frac{\partial E}{\partial v_y} = 0$$

$$\frac{\partial}{\partial x} \left(\frac{\partial E}{\partial u_x} \right) = \frac{\partial}{\partial x} \left(\frac{\partial}{\partial u_x} (\alpha u_x^2) \right) = 2\alpha u_{xx}$$

$$\begin{aligned} \textcircled{1} \quad \frac{\partial E}{\partial u} - \frac{\partial}{\partial x} \frac{\partial E}{\partial u_x} - \frac{\partial}{\partial y} \frac{\partial E}{\partial u_y} &= 2(I_x u + I_y v + I_t) I_x - 2\alpha u_{xx} - 2\alpha u_{yy} \\ &= 2(I_x u + I_y v + I_t) I_x - 2\alpha \Delta u = 0 \end{aligned}$$

$$\text{where : } \Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = u_{xx} + u_{yy}$$

Laplace operator

Hence :

$$2(I_x u + I_y v + I_t) I_x - 2\alpha \Delta u = 0$$

$$2(I_x u + I_y v + I_t) I_y - 2\alpha \Delta v = 0$$

$$Q: \Delta u = U_{xx} + U_{yy}$$

How to compute Δu from an image?

$$A: U_{xx} = \frac{\partial^2 u}{\partial x^2}$$

$$= \frac{\partial}{\partial x} \left(\frac{\partial u}{\partial x} \right) = \frac{\partial}{\partial x} (U(x+1, y) - U(x, y))$$

$$= (U(x+1, y) - U(x, y)) - (U(x, y) - U(x-1, y))$$

$$= (U(x+1, y) + U(x-1, y)) - 2U(x, y)$$

$$U_{xx} = (U(x+1, y) + U(x-1, y)) - 2U(x, y)$$

$$U_{yy} = (U(x, y+1) + U(x, y-1)) - 2U(x, y)$$

$$\frac{1}{4} \Delta u = \frac{1}{4} (U_{xx} + U_{yy})$$

$$= \frac{1}{4} \left[U(x+1, y) + U(x-1, y) + U(x, y+1) + U(x, y-1) \right] - U(x, y)$$

average of neighboring pixels of (x, y)

$$\frac{1}{4} \Delta u(x, y) = \overline{U(x, y)} - U(x, y)$$

local average

Recall:

$$(I_x u + I_y v + I_t) I_x - \alpha \Delta u = 0$$

$$(I_x u + I_y v + I_t) I_y - \alpha \Delta v = 0$$



$$(I_x u + I_y v + I_t) I_x - \alpha' (\bar{u} - u) = 0 ; \quad \alpha' = \alpha/4$$

$$(I_x u + I_y v + I_t) I_y - \alpha' (\bar{v} - v) = 0$$



Since α is decided by us
we can also claim $\alpha = \alpha'$

$$\begin{bmatrix} \alpha + I_x^2 & I_x I_y \\ I_y I_x & \alpha + I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \alpha \bar{u} - I_x I_t \\ \alpha \bar{v} - I_y I_t \end{bmatrix}$$

We can't solve it, as we don't know \bar{u} & \bar{v}

$$\begin{bmatrix} \alpha + I_x^2 & I_x I_y \\ I_y I_x & \alpha + I_y^2 \end{bmatrix} \begin{bmatrix} U \\ V \end{bmatrix} = \begin{bmatrix} \alpha \bar{U} - I_x I_t \\ \alpha \bar{V} - I_y I_t \end{bmatrix}$$

For 1 pixel: $\underset{2 \times 2}{A} \underset{2 \times 1}{\bar{w}} = \underset{2 \times 1}{\bar{b}}$

For N pixels: $\underset{2N \times 2N}{A} \underset{2N \times 1}{\bar{w}} = \underset{2N \times 1}{\bar{b}}$

At this point, we can use any method to solve the equation. Yet, we should consider that A is a huge matrix and \bar{U}, \bar{V} in \bar{b} are unknown.

e.g. $N=2$

$$A = \begin{bmatrix} \alpha + I_{x,1,1}^2 & I_{x,1,1} I_{y,1,1} & 0 & 0 \\ I_{y,1,1} I_{x,1,1} & \alpha + I_{y,1,1}^2 & 0 & 0 \\ 0 & 0 & \alpha + I_{x,1,2}^2 & I_{x,1,2} I_{y,1,2} \\ 0 & 0 & I_{y,1,2} I_{x,1,2} & \alpha + I_{y,1,2}^2 \end{bmatrix}; \quad \bar{w} = \begin{bmatrix} U_1 \\ V_1 \\ U_2 \\ V_2 \end{bmatrix}$$

• Further solution proposed by Horn & Schunck:

Considering each pixel (i, j) :

$$\begin{bmatrix} \alpha + I_{x,ij}^2 & I_{x,ij} I_{y,ij} \\ I_{x,ij} I_{y,ij} & \alpha + I_{y,ij}^2 \end{bmatrix} \begin{bmatrix} U_{ij} \\ V_{ij} \end{bmatrix} = \begin{bmatrix} \alpha \bar{U}_{ij} - I_{x,ij} I_t \\ \alpha \bar{V}_{ij} - I_{y,ij} I_t \end{bmatrix}$$

$$\begin{bmatrix} U_{ij} \\ V_{ij} \end{bmatrix} = \underbrace{\begin{bmatrix} \alpha + I_{x,ij}^2 & I_{x,ij} I_{y,ij} \\ I_{x,ij} I_{y,ij} & \alpha + I_{y,ij}^2 \end{bmatrix}^{-1}}_{A} \begin{bmatrix} \alpha \bar{U}_{ij} - I_{x,ij} I_t \\ \alpha \bar{V}_{ij} - I_{y,ij} I_t \end{bmatrix}$$

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \rightarrow A^{-1} = \frac{1}{\det(A)} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$\begin{aligned} \det(A) &= (\alpha + I_{x,ij}^2)(\alpha + I_{y,ij}^2) - I_{x,ij}^2 I_{y,ij}^2 \\ &= \alpha^2 + \alpha I_{x,ij}^2 + \alpha I_{y,ij}^2 \end{aligned}$$

$$\begin{bmatrix} U_{ij} \\ V_{ij} \end{bmatrix} = \frac{1}{\alpha^2 + \alpha I_{xij}^2 + \alpha I_{yij}^2} \begin{bmatrix} \alpha + I_{yij}^2 & -I_{xij} I_{yij} \\ -I_{xij} I_{yij} & \alpha + I_{xij}^2 \end{bmatrix} \begin{bmatrix} \alpha \bar{U}_{ij} - I_{xij} I_{tij} \\ \alpha \bar{V}_{ij} - I_{yij} I_{tij} \end{bmatrix}$$

$$U_{ij} = \frac{(\alpha + I_{yij}^2)(\alpha \bar{U}_{ij} - I_{xij} I_{tij}) - (I_{xij} I_{yij})(\alpha \bar{V}_{ij} - I_{yij} I_{tij})}{\alpha^2 + \alpha I_{xij}^2 + \alpha I_{yij}^2}$$

$$= \frac{\cancel{\alpha^2} \bar{U}_{ij} - \cancel{\alpha} I_{xij} I_{tij} + \cancel{\alpha} I_{yij}^2 \bar{U}_{ij} - \cancel{\alpha} I_{xij} I_{yij} \bar{V}_{ij}}{\cancel{\alpha} (\alpha + I_{xij}^2 + I_{yij}^2)}$$

\downarrow
 \downarrow

$$(\alpha + I_{xij}^2 + I_{yij}^2) U_{ij} = (\alpha + I_{yij}^2) \bar{U}_{ij} - I_{xij} I_{yij} \bar{V}_{ij} - I_{xij} I_{tij}$$

$$(\alpha + I_{xij}^2 + I_{yij}^2) U_{ij} - (\alpha + I_{yij}^2) \bar{U}_{ij} = - (I_{xij} I_{yij} \bar{V}_{ij} + I_{xij} I_{tij})$$

$$(\alpha + I_{xij}^2 + I_{yij}^2) U_{ij} - (\alpha + I_{yij}^2 + I_{xij}^2) \bar{U}_{ij} = - I_{xij} (I_{yij} \bar{V}_{ij} + I_{tij}) - I_{xij}^2 \bar{U}_{ij}$$

$$(\alpha + I_{xij}^2 + I_{yij}^2) U_{ij} = (\alpha + I_{xij}^2 + I_{yij}^2) \bar{U}_{ij} - [I_{xij} (I_{yij} \bar{V}_{ij} + I_{xij} \bar{U}_{ij} + I_{tij})]$$

Applying the same operations to V_{ij} :

$$(\alpha + I_{xij}^2 + I_{yij}^2) \begin{bmatrix} U_{ij} \\ V_{ij} \end{bmatrix} = \begin{bmatrix} (\alpha + I_{xij}^2 + I_{yij}^2) \bar{U}_{ij} - [I_{xij} (I_{yij} \bar{V}_{ij} + I_{xij} \bar{U}_{ij} + I_{tij})] \\ (\alpha + I_{xij}^2 + I_{yij}^2) \bar{V}_{ij} - [I_{yij} (I_{yij} \bar{V}_{ij} + I_{xij} \bar{U}_{ij} + I_{tij})] \end{bmatrix}$$

Iterative solution using the Gauss-Seidel method:

$$A\bar{x} = \bar{b} \quad ; \quad A = L + D + U$$

then:

$$\hat{x}^{\text{new}} = (L + D)^{-1} (\bar{b} - U \bar{x}^{\text{old}})$$

$$A = \begin{bmatrix} \alpha + I_{xij}^2 + I_{yij}^2 & 0 \\ 0 & \alpha + I_{xij}^2 + I_{yij}^2 \end{bmatrix} \quad ; \quad \begin{array}{l} L = 0 \\ D = A \\ U = 0 \end{array}$$

$$\bar{x} = \begin{bmatrix} U_{ij} \\ V_{ij} \end{bmatrix}$$

Hence :

$$\bar{x}^{\text{new}} = (\mathbb{I} + \mathbb{D})^{-1} (\bar{b} - \mathbb{D} \bar{x}^{\text{old}}) \quad ; \quad \mathbb{I} = \mathbb{D} = \mathbb{O}$$
$$= \mathbb{D}^{-1} \bar{b}$$

$$\mathbb{D} = \begin{bmatrix} \alpha + I_{xij}^2 + I_{yij}^2 & 0 \\ 0 & \alpha + I_{xij}^2 + I_{yij}^2 \end{bmatrix} \rightarrow \mathbb{D}^{-1} = \frac{1}{\alpha + I_{xij}^2 + I_{yij}^2}$$

$$U_{ij}^{\text{new}} = \bar{U}_{ij}^{\text{old}} - \frac{I_{xij} (I_{xij} \bar{U}_{ij}^{\text{old}} + I_{yij} \bar{V}_{ij}^{\text{old}} + I_{tij})}{\alpha + I_{xij}^2 + I_{yij}^2}$$

$$V_{ij}^{\text{new}} = \bar{V}_{ij}^{\text{old}} - \frac{I_{yij} (I_{xij} \bar{U}_{ij}^{\text{old}} + I_{yij} \bar{V}_{ij}^{\text{old}} + I_{tij})}{\alpha + I_{xij}^2 + I_{yij}^2}$$

Final equations proposed by Horn & Schunck

Note :

$$\begin{bmatrix} \alpha^2 + I_x^2 & I_x I_y \\ I_x I_y & \alpha^2 + I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \alpha^2 \bar{u} - I_x I_t \\ \alpha^2 \bar{v} - I_y I_t \end{bmatrix}$$

$$A \bar{w} = \bar{b}$$

For N pixels: $(2N \times 2N) \times (2N \times 1)$

where :

A is a matrix whose elements are known.

\bar{w} is a vector containing (u, v) that are unknown.

\bar{b} is a vector containing (\bar{u}, \bar{v}) that are also unknown.

Q: Can we assume \bar{u}^{old} & \bar{v}^{old} to get vector \bar{b} ?

A: One possibility is to initialize u & v maps with zero, and then compute \bar{b} .

Having computed \bar{b} , we obtain $\bar{w} = A^{-1} \bar{b} (w_{init})$

and do this iteratively : $\bar{w}_{new} = A^{-1} \bar{b} (\bar{w}_{old})$.

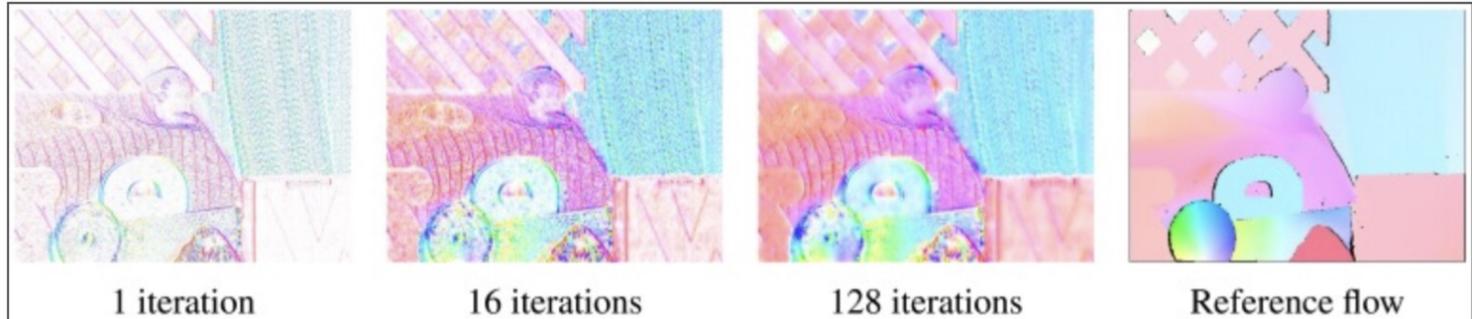


However, computing A^{-1} in every update is expensive, since if we have a 100×100 image $\rightarrow N = 10,000$, thus the dimension of $A = 20,000 \times 20,000$.

[6] Secrets of Robust Optic Flow

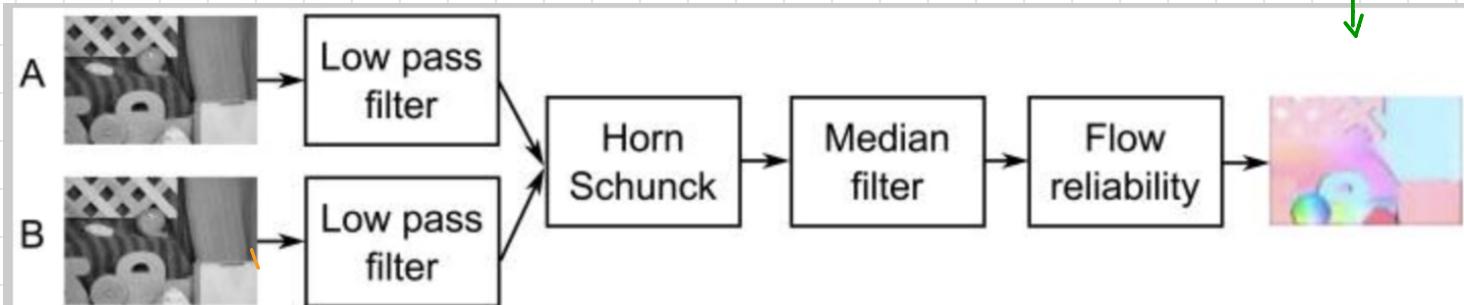
#13

Optical flow results using the HS method:



It works, but the results are noisy & inaccurate.

Below is one of the possible ways to improve:

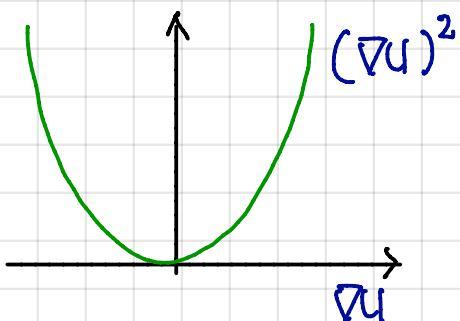


Q: What are the secrets for robust flow estimation?

- A:
- (1) Robust objective function
 - (2) Coarse to fine mechanism
 - (3) Interpolation
 - (4) Median Filtering
 - (5) Pre-processing
 - (6) Gradient Constancy Constraint

[7] Robust Objective Functions #14

Problem: A quadratic function is convenient to optimize, but it is statistically not robust.



This curve means: when ∇u is larger the penalty, $(\nabla u)^2$, will be higher.



It's indeed something we want.



However, when u suffers from noise, inevitably the penalty is also high; which is not good, since the actual penalty should be small.

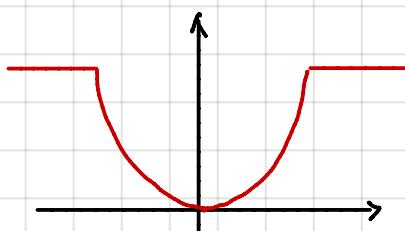
Since, without noise, that value of u is the correct one.



In other words, we penalize a correct value of u with a high cost because it suffers from noise.

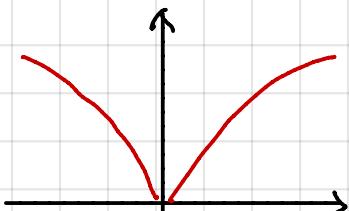
More robust functions:

(1) Truncated Quadratic:



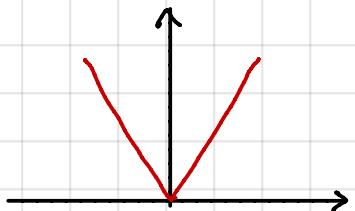
$$f(x) = \begin{cases} \lambda x^2 & \text{if } |x| < \frac{\sqrt{\lambda}}{\sqrt{\alpha}} \\ \alpha & \text{else} \end{cases}$$

(2) Lorentzian



$$f(x) = \log \left(1 + \frac{1}{2} \left(\frac{x}{\delta} \right)^2 \right)$$

(3) Charbonnier



$$f(x) = (x^2 + \varepsilon^2)^{\alpha}$$

$$\alpha = 0.5$$

if $\alpha < 0.5$ the function becomes non-convex.

[8] Coarse to Fine

#IS

Problem: Original HS method doesn't work for large displacement or large motion.

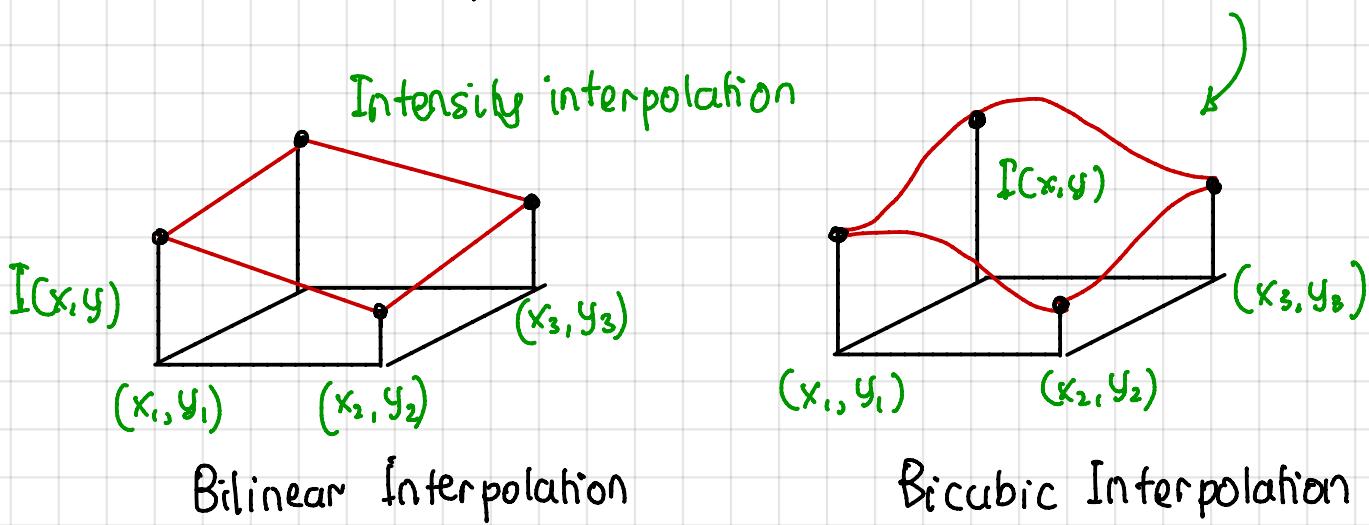
Solution:

- (1) Perform a standard multi-resolution technique [to generate pyramid images]
- (2) Estimate optic flow at a coarse level
- (3) For the next finer level, warp the second image to the first image using the estimated optic flow in step #2.
- (4) Estimate optic flow from the warped images 2 & 1.
- (5) Do step #2 to #4 till the finest level.

[ii] Interpolation

Problem: The optic flow from the coarser level is more sparse than the number of pixels in the finer level.

Solution: Interpolation \rightarrow Bicubic interpolation



[g] Median Filtering

Problem: Optic flow estimation can be noisy, and the median filtering can smooth out the noise.

Median filter: To replace each point or pixel with the median of the neighboring points (5×5 pixels)



Solution: To apply the median filtering to the intermediate flow after every warping iteration.

[10] Gradient Constancy Constraint

Problem: The BCC is not robust to the light changes

Solution: Gradient constancy assumption (GCC):

$$\nabla I(x, y, t) = \nabla I(x+u, y+v, t+1)$$

$$\begin{bmatrix} \frac{\partial I(x, y, t)}{\partial x} \\ \frac{\partial I(x, y, t)}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial I(x+u, y+v, t+1)}{\partial x} \\ \frac{\partial I(x+u, y+v, t+1)}{\partial y} \end{bmatrix}$$

$$\frac{d \nabla I(x, y, t)}{dt} = 0$$

GCC

$$\frac{d}{dt} \begin{bmatrix} I_x \\ I_y \end{bmatrix} = \begin{bmatrix} \frac{\partial I_x}{\partial x} \frac{\partial x}{\partial t} + \frac{\partial I_x}{\partial y} \frac{\partial y}{\partial t} + \frac{\partial I_x}{\partial t} \\ \frac{\partial I_y}{\partial x} \frac{\partial x}{\partial t} + \frac{\partial I_y}{\partial y} \frac{\partial y}{\partial t} + \frac{\partial I_y}{\partial t} \end{bmatrix} = \underbrace{\begin{bmatrix} I_{xx} u + I_{xy} v + I_{xt} \\ I_{yx} u + I_{yy} v + I_{yt} \end{bmatrix}}_{=0}$$

We can combine the BCC & GCC in one objective function.

Combining the BCC & GCC :

#17

$$E(u, v) = (I_x u + I_y v + I_t)^2 \xrightarrow{\text{BCC}}$$

$$+ \frac{1}{2} \left\| \begin{array}{l} I_{xx} u + I_{xy} v + I_{xt} \\ I_{yx} u + I_{yy} v + I_{yt} \end{array} \right\|_2^2 + \alpha (\|\nabla u\|_2 + \|\nabla v\|_2)$$

GCC

Regularization